RECEIVED
CENTRAL FAX CENTER
JUL 24 2006

AMENDMENTS TO THE SPECIFICATION

Please replace paragraph beginning on page 11, line 19 with the following amended paragraph:

Fig. 2 is a schematic diagram of a fetching algorithm in a multistreaming architecture. The multistreaming architecture corresponds to a processor having multiple hardware streams STREAM 1-STREAM S supporting multiple data threads, a data cache (not shown). The schematic diagram shows a system according to the present invention for fetching instructions from one to P of the multiple hardware streams STREAM 1-STREAM S, where P is less than the number of multiple hardware streams STREAM 1-STREAM S. The diagram shows multiple hit/miss predictors PRED 1-PRED S, each associated with a corresponding one of the multiple hardware streams STREAM 1-STREAM S, and each configured to forecast whether corresponding instructions from the corresponding one of the multiple hardware streams STREAM 1-STREAM S will hit or miss the data cache, where the multiple hit/miss predictors PRED 1-PRED S forecast whether the corresponding instructions from the corresponding one of the multiple hardware streams will hit or miss the data cache prior to when the corresponding instructions enter into a dispatch stage (not shown) in the pipeline. The diagram also shows a fetch stage that is coupled to the multiple hit/miss predictors PRED 1-PRED S. The fetch stage is configured to simultaneously fetch every cycle, the instructions from the one to P of the multiple hardware streams STREAM 1-STREAM S to the pipeline. The fetch stage is additionally configured to select, on a cycle-by-cycle basis, the one to P of the multiple hardware streams STREAM 1-STREAM S from which to fetch the instructions. The diagram also depicts an instruction scheduler that is coupled to the fetch stage. The instruction scheduler manages access for the multiple hardware streams STREAM 1-STREAM S to a set of functional resources (not shown) for processing instructions from the multiple hardware streams STREAM 1-STREAM S, where at any point in time, said instruction scheduler manages access for a given one of the multiple hardware streams STREAM 1-STREAM S according to a priority record within a priority file, regardless of any priority associated with the multiple data threads. The fetch stage

includes a fetch algorithm. The algorithm decides from which stream(s) STREAM 1-STREAM S to fetch based on cache hit/miss predictors associated to each of the streams STREAM 1-STREAM S. In Fig. 2 a predictor PRED 1-PRED Sis associated with streams 1, 2, and so on through stream S. Thus, theoretically, instructions from up to S streams STREAM 1-STREAM S (S being the maximum number of streams STREAM 1-STREAM S supported by the multistreaming architecture) can be simultaneously fetched every cycle. In reality, however, the fetching algorithm might be restricted to fetch instructions from P streams (P<S) due to implementation restrictions (for example, availability of instruction cache ports). Moreover, the fetching algorithm might select from which streams to fetch based on other information (for example, confidence on the branch predication of each stream, thread priorities, state of the pipeline, etc.)